# Appendix G - MIME and Mapping

## What is MIME?

“MIME” stands for Multipurpose Internet Mail Extensions. MIME serves two major purposes – it allows mail applications to tell one another what sort of data is in mail, and it also provides standard ways for mail applications to encode data so that it can be sent through the Internet mail system.

## MIME Encodings

The Internet uses the “SMTP” protocol to move mail around. SMTP is limited to the US-ASCII character set (see Appendix E). This is a problem for people who speak languages other than American English and so need accented characters or non-American letters, or for people who want to use special symbols like section mark (§).

MIME provides a way around this restriction. It offers two encodings, “quoted-printable” and “base64.” These encodings use US-ASCII character codes to represent any sort of data you like, including special characters or even non-text data.

“Quoted-printable” is used for data that is mostly text, but has special characters or very long lines. It’s very simple. Quoted-printable looks just like regular text, except when a special character is used. The special character is replaced with an “=”and two more characters that represent the character code of the special character. So, a section mark (§) in quoted-printable looks like “=A8”.

However, there are some other things that quoted-printable does. For one, since it uses an “=” to mean something special, equal signs must themselves be encoded (as “=3D”). Second, no line in quoted-printable is allowed to be more than 76 characters long. If your mail has a line longer than 76 characters, the quoted-printable encoding will break your line in two and put an “=” at the end of the first line, to signal to the mail reader at the other end that the two lines are really supposed to be all one line. Finally, a few mail systems either add or remove spaces from the ends of lines. So, in quoted-printable, any space at the end of a line gets encoded (as “=20”), to protect it from such mail systems.

Let’s try an example. Here’s a passage of text that you might type on your Macintosh:

```
«Il est démontré, disait-il, que les choses ne peuvent être
autrement; car tout étant fait pour une fin, tout est nécessairement
pour la meilleure fin.»
```

Without any encoding, this might show up on your recipient’ s screen as:

```
+Il est dimontri, disait-il, que les choses ne peuvent btre
autrement; car tout itant fait pour une fin, tout est nicessairement
pour la meilleure fin.;
```

This corruption happens because SMTP cannot handle the special characters. However, ifyou and your recipient both have MIME, quoted-printable encoding would be used, and your text would show up properly:

```
«Il est démontré, disait-il, que les choses ne peuvent être
autrement; car tout étant fait pour une fin, tout est nécessairement
pour la meilleure fin.»
```

While your mail was actually in transit, however, it would have looked like:

```
=ABIl est d=E9montr=E9, disait-il, que les choses ne peuvent =EAtre
=
autrement; car tout =E9tant fait pour une fin, tout est n=E9cessairement = .
pour la meilleure fin.=BB
```

Base64 encoding is another way to protect binary data from the SMTP mail system. However, Base64 makes noattempt tobe legible, and is most appropriate for non-text data.

## MIME Labeling

The other important part of MIME is that it lets mailers communicate what kind of data is in a message (or part of a message). The primary mechanism used for this is the Content-Type header:

Content-Type: text/plain; charset=iso-8859- 1

A content-type header is divided into three parts; the content type, the content subtype, and the parameters. In this case, the content type is "text," meaning the message contains mostly legible text. The content subtype is "plain," which means there aren' t any formatting commands or anything like that embedded in the text. Finally, "charset=iso-8859- 1" is a parameter; in this case it identifies the character set the message uses.

The major content types are:

| | |
|---|---|
| text | legible text |
| image | pictures and graphics |
| audio | sound |
| video | moving pictures |
| message | messages or pieces of messages |
| multipart | several different kinds of data in a single message |

## Practical Issues

There are really only two things you sometimes need to do with Eudora and MIME. One is that it may occasionally be necessary to turn off quoted-printable encoding. Another is that you may want to know how to define mappings between MIME types and Macintosh types.

### Turning Off Quoted-Printable

Eudora automatically uses quoted-printable encoding if your mail contains special characters. Eudora also uses quoted-printable encoding for attached plain text files. If your recipients don't have MIME, quoted-printable can hurt more than it helps. If that's the case, just turn off the QP icon when you are sending text files to those recipients.

## Mapping Between MIME Types and Macintosh Types

When you send attached files to other Eudora users, Eudora automatically knows what kind of data is in the files, because Eudora sends along special information with the file. However, if you're sending the file to a non-Macintosh user, or receiving files from a non-Macintosh user, it's important to get the right MIME type information on the file, or for Eudora to understand what the MIME type information means.

Eudora knows about some MIME types. However, since new MIME types are being defined all the time, it may be necessary to add to Eudora's knowledge from time to time. If you're familiar with ResEdit, this isn't too hard to do.

The way Eudora maps between MIME and Macintosh types is with EuIM and EuOM resources. EuOM resources are used for sending attachments, EuIM for receiving. They have the same basic structure.

EuOM and EuIM resources are lists of individual elements called "maps." Each map describes a Macintosh document type (or MIME data type) and then lists what MIME data type (or Macintosh document type) it corresponds to. For any given type, Eudora looks through all the maps in all the EuOM or EuIM resources, and uses the best match.

*Note: EuOM and EuIM resources are also used when uuencoding and uudecoding files, so that filename suffixes can be mapped to and from Macintosh types. A good set of EuIM and EuOM resources can substantially improve document exchange with systems that use uuencode.*

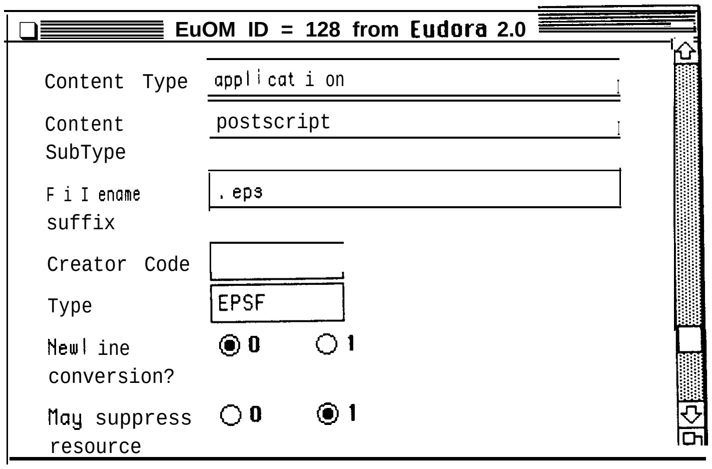
### Sending

When you create a map in an EuOM resource, you use the "Creator Code" and "Type" fields to specify what documents the map applies to. These fields should be filled with the four-byte creator code or Macintosh type of the documents you want to send. If you leave the Creator Code blank, but fill in the type, the map is used for any document of that type, regardless of creator. If you fill in both Creator Code and Type, a document has to match both for the map to be used. Given the choice, Eudora uses the map that matches both creator and type.

The other parts of the map are used to construct the MIME information. Content Type and Content Subtype are the MIME type and subtype to use for the document. Filename suffix allows you to tell Eudora to add a suffix to the filename, as an extra hint to the receiving system (for example, you might have Eudora add ".xls" to Excel files).

"Newline conversion?" tells Eudora whether or not to convert carnage returns in the file to carriage return, linefeed. Usually, you should set this to 1 for text data, but to O for binary files.

Finally, "May suppress resource fork?" is used in conjunction with Eudora's Always include Macintosh information option. If you set this to 1, and Always include Macintosh information is off, Eudora won' t send Macintosh type and creator information with the file, and won' t send the resource fork. Instead, it will just send the data fork with the MIME information attached to it.


EuOM ID = 128 from Eudora 2.0

| | | |
|---|---|---|
| Content Type | appl i cat i on | |
| Content SubType | postscript | |
| F i l ename suffix | . eps | |
| Creator Code | | |
| Type | EPSF | |
| Newl ine conversion? | [✓ 0] | 1 |
| May suppress resource | 0 | [✓ 1] |

*An Example Map in an EuOM Resource*

The map above says that all files of type "EPSF," no matter what the creator, should be sent as "application./postscript," that ".eps" should be added to the filename, that carriage returns should not be turned into carriage return/linefeed pairs, and that when the Always As Documents switch is off, the resource fork won' t be sent.

## Receiving

EuIM resources are used for receiving files. They're pretty much the same as EuOM resources, except that the MIME type and subtype are used for matching, and the Macintosh creator code and type are applied to the file received.

As with EuOM resources, you can leave parts blank. If you want to match all files with an ".eps" suffix, regardless of the MIME type or subtype, leave the type and subtype blank. If you don' t care what the filename suffix is, leave that blank and match with the MIME type and/or subtype only. Again, as with EuOM resources, Eudora will choose the map that matches best.

With EuIM resources, it's sometimes a good idea to use several maps to catch all important cases. For example, it might be a good idea to have three maps for dealing with PostScript files, as follows:

Content Type: application
Content Subtype: postscript
Filename suffix:
Creator Code: mlpr
Type: TEXT

This map will catch most MIME PostScript files, and set their creator to MacLPR.

Content Type:
Content Subtype:
Filename suffix: .eps
Creator Code: dPro
Type: EPSF

This map will match any incoming file with a suffix of ".eps," regardless of the MIME type info, and set it's type to "EPSF" and creator to "dPro" (MacDraw Pro). But what if a file comes in with a suffix of ".eps" and a MIME type/subtype of "application/postscript"? Which map gets used? The first one gets used; when Eudora has a choice between matching a suffix and matching MIME type information, MIME wins. A third map may be in order:

Content Type: application
. Content Subtype: postscript
Filename suffix: .eps
Creator Code: dPro
Type: EPSF

This makes application/postscript files with suffixes of ".eps" get type EPSF and creator dPro.